Contents lists available at ScienceDirect

INTEGRATION, the VLSI journal

journal homepage: www.elsevier.com/locate/vlsi

Fast and accurate FPGA-based framework for processor architecture vulnerability analysis

Hoda Mahdiani, Saeed Safari, Mostafa E. Salehi *

School of Electrical and Computer Engineering, University of Tehran, Tehran 14395-515, Iran

ARTICLE INFO

Article history:
Received 7 June 2015
Received in revised form
25 November 2015
Accepted 25 November 2015
Available online 2 December 2015

Keywords:
Architecture Vulnerability analysis
Soft error
Embedded processors
Fault modeling and emulation
Field-programmable gate array

ABSTRACT

This paper presents a fast, accurate, and flexible FPGA-based fault emulation platform, namely FARAVAM that can be exploited for AVF analysis in modern microprocessors. The proposed approach provides fault injection capabilities supporting automatic modification of post-synthesis net-lists and introduces a highly controllable and observable transient fault analysis environment. The presented vulnerability analysis platform using both exhaustive and random fault emulation approaches, provides useful information for identifying areas threatening reliability to make processors more fault tolerant. We applied our platform for extracting the best trade-offs between precision and speed up in vulnerability analysis of MIPS processor. The experimental results indicate that in addition to having high precision we obtain about seven orders of magnitude speed up in comparison with simulation based vulnerability analysis techniques.

1. Introduction

Reliability as well as correctness of operations are significant considerations in the design and verification of modern computer systems and advanced microprocessors. While on the other hand, soft errors are among the most effective factors in reducing system reliability. A soft error is caused by a transient pulse that changes the logical state of a vulnerable semiconductor device. This transient pulse might be created by a high energy particle hitting to some critical points of a circuit, or by others factors such as voltage drops in power supply network, temperature fluctuations, and gate length and doping concentration variations [1–3]. While both of the combinational and sequential logic elements of a microprocessor are vulnerable to soft errors, the AVF factor for a specific hardware structure is defined as the probability of a soft fault leading to an architecturally observable error and changing the output of the running program [4].

AVF analysis of a digital VLSI chip is very important for understanding the behavior of the system in terms of reliability and fault masking capabilities. Therefore, various categories of AVF measurement and analysis methods are presented in hardware structures and microprocessor systems. These methods can be generally classified into two categories of static and dynamic approaches. In static methods, it is not necessary to simulate the processor behavior in runtime. Instead, only some statically defined properties of a system are used to estimate the AVF of a hardware structure. In dynamic methods, some extracted information is needed for AVF analysis through simulation of processor operations in runtime [6]. In another classification, AVF analysis methods can be divided into three classes of analytical models, performance models and statistical fault injection mechanisms [7].

Analytical models calculate AVF using the average number of architecturally correct execution (ACE) bits in a hardware structure [4,5] and [7]. This method is not accurate but is fast and practical and so can be used in early stages of design when an RTL model is not available. Performance models use some defined rules to estimate AVF based on the fraction of time that a bit in the structure is ACE or Un_ACE in its life time [4,5] and [8]. This technique is fast and can compute the AVFs of many structures in parallel while its main drawback is that it requires sufficient details about architecture and microarchitecture of the circuit under analysis.

AVF extraction using fault injection techniques is the most widely accepted and adopted approach in various fields of reliability analysis and circuit testing. In this method, some randomly generated bit flips are injected to RTL or gate-level designs when the workload execution is in progress [9–11]. The propagation of this error to the outputs of the circuit is then observed. In this case, AVF is calculated as the ratio of the number of observed mismatches (between outputs of faulty and golden models of design) to the total number of injected faults [6].

* Corresponding author.
E-mail addresses: hoda.mahdiani@ut.ac.ir (H. Mahdiani), saeed@ut.ac.ir (S. Safari), mersali@ut.ac.ir (M.E. Salehi).

http://dx.doi.org/10.1016/j.vlsi.2015.11.005
0167-9260/© 2015 Elsevier B.V. All rights reserved.

Literature survey in the domain of fault injection determines that these techniques are classified in *hardware*, *software, simulation-based* and *emulation-based fault injection* methods [12–15].

In hardware based methods faults are imposed to the actual structure of system under test by artificially generating some disturbing environmental parameters [10], and [16–19]. The speed of experiments in actual situation is very fast but in these methods controllability and observability of fault propagation to internal parts of a circuit is limited. Also the main circuit may be damaged.

In software based methods faults are injected in software level and memories of design. These software faults can be the impact of hardware faults [20–23]. These methods are low cost because no hardware is required for fault injection and they do not have any effect on the main circuit. However, the ability of control and observation of hardware fault effects are dependent on the amount of software access to hardware.

Simulation based techniques are those that fault injection is done in HDL description of a hardware structure during the simulation of its behavior [14,24–28]. These methods are applicable in every phase of the system generation and provide good controllability and observability of internal circuit points. However, the simulation of these fault injection experiments is very time consuming.

The other approach that attracts fault injection studies in recent years includes methods based on using FPGAs and their facilities which are called FPGA-based techniques. In these methods, faults are injected to the implemented circuit structure on an FPGA chip. These techniques support the flexibility and high ability of simulation based methods in the field of fault injection control and observation, whereas they have the efficiency and speed of hardware based techniques [26,29–39]. These methods can be used in every phase of the system design and can contribute in removing the bugs of designed system before manufacturing the original plan. In other words, a designer can perceive the behavior of a system in actual situations whereas these approaches will not harm the physical implemented instances of the designed systems and hence, reducing the cost of fault injection experiments.

In previous studies FPGA-based fault injections were mostly used in permanent stuck at fault injections in digital systems for generating a suitable set of test vectors for developing circuit test capabilities [29,36]. However, the injection of transient faults during operational mode of the systems and estimation of the system vulnerability and fault tolerance with FPGA-based methods are added to recent studies [34,35,37–40].

In this paper we develop a framework consisting of a tool chain for AVF analysis of embedded processors by design and implementation of a dynamic platform for transient fault emulation using the facilities of FPGA-based methods.

The reminder of this paper is organized as follows. Section 2 describes the fault model of the represented fault emulation platform. In Section 3 the emulation platform and the proposed framework for processor AVF analysis are described. Then experimental results are presented in Section 4 and Section 5 concludes the paper.

2. Fault model

Our presented platform has the ability of injecting different types of faults such as transient and permanent stuck-at faults. However, the contribution of this paper is the method of measuring the architectural vulnerability of microprocessor structures. Since models of permanent faults are simpler than transient faults, we will focus on transient faults as the basic model and other fault models are just mentioned.

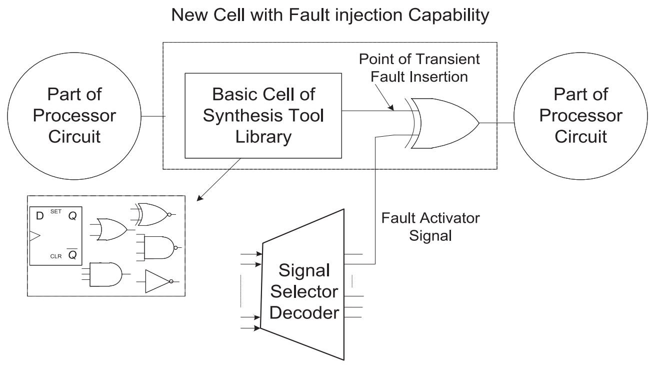
New Cell with Fault injection Capability
Part of Processor Circuit
Basic Cell of Synthesis Tool Library
Point of Transient Fault Insertion
Part of Processor Circuit
D
SET
Q
CLR
Q
Fault Activator Signal
Signal Selector Decoder

Fig. 1. Adding capability of transient fault injection to basic cells of a standard synthesis tool library.

Our approach belongs to the category of instrumentation-based methods. These types of methods are popular in both fault simulation and fault emulation frameworks and can be applied in different abstraction levels such as RTL or gate. The key concept is that a saboteur element is added to the target component or signal in order to emulate fault injection. On the other hand, a custom fault injector component simulates the behavior of fault model on the victim signal of the structure whereas the fault injector component is inactive when the fault free functionality of system under evaluation is expected.

To achieve higher controllability, observability, and accuracy, fault injector elements are inserted in post-synthesis ASIC net-list of the CPU circuit by using a parser. The parser is developed for automatically annotating and replacing all basic gate level components of CPU net-list with new cells. These new cells support all functionalities of the basic cells in addition to having an extra input signal which controls fault injection to cell outputs in every cycle of processor operation.

As shown in Fig. 1, for transient faults the output of the basic cell is XORed with the logic of control signal for flipping the selected victim signal of the CPU. Bit flip is the most commonly used technique for modeling transient faults in literature. For permanent stuck at 1 fault modeling, an OR gate is used instead of XOR cells and for stuck at 0 fault model, XOR cell is replaced with an AND gate with inverted input control signal.

In this method control signals have a critical role in fault injection mechanism. Due to the increasing complexity of digital designs, control signals must be managed properly for calculating the vulnerability of advanced processor structures. Distributed decoders or shift registers are apt choices for fault injection activation. In our approach, for single event upset (SEU) and single event transient (SET) faults, decoders are used to inject a fault in a single cycle in the target position of the circuit. An external controller in fault emulation platform is designed that is responsible for managing all complexities in fault injection activation and determines which faults should be activated in which CPU cycles of workload execution.

3. AVF analysis framework

Our proposed FPGA-based AVF analysis and fault emulation framework has been named FARAVAM (Flexible, Accurate and Rapid Architecture Vulnerability Analysis in Microprocessors). It offers a method for fast and accurate evaluation of application fault masking and vulnerability of an ASIC design. Some features and capabilities provided by our proposed FPGA based AVF analysis framework, are described below.